|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| logowydzialu | Instytut Informatyki Politechniki Śląskiej  Zespół Mikroinformatyki i Teorii Automatów Cyfrowych | | | logoii | |
| **Rok akademicki:** | **Rodzaj studiów\*: SSI/NSI/NSM** | **Przedmiot:** | | **Grupa** | **Sekcja** |
| **2018/2019** | **SSI** | **BIAI** | | **GKiO3** | **7** |
| **Autor:** | **Aleksandra Musiał (alekmus576@student.polsl.p)** | **Prowadzący**: | | **GB** | |
| **Autor:** | **Benedykt Waszkiewicz (benewas355@student.polsl.pl)** |
| ***Raport końcowy*** | | | | | |
| **Temat projektu:**  Classification of the genre of the film based on the description of the plot | | | | | |
| **Data oddania:**  **dd/mm/rrrr** | | |  | | |

1. Introduction

The aim of our project was to prepare a program capable of recognizing the genre of the movie based on the plot using neural network. We decided to implement the network in Python and use TensorFlow library.

1. Data set

We used dataset from www.kaggle.com which contains information about 35 000 movies. There are following columns in the set:

- release year,

- title,

- ethnicity,

- director,

- cast,

- genre,

- Wikipedia page,

- plot.

The release year varies from 1901 to 2017. As for genres, there are about 20 main genres but often multiple genres are assigned to one movie. Because of the assumptions we made at the beginning, we cut the dataset and left only columns with a genre ant a plot.

1. Data preprocessing

/\* TO DO \*/

1. Implementation
   1. Keras

Keras is one of the leading high-level neural networks APIs. It is written in Python. The core data structure in Keras is the model which is available in two options – sequential and the Model class used with the functional API. We prepared sequential model.

from tensorflow import keras

model = keras.Sequential()

* 1. Layers

We started building our neural network with the embedding layer. It is responsible for taking vocabulary in numerical form and looking up the embedding vector for each word-index. These vectors are learned as the model trains. It results in additional dimension in the output array – batch, sequence and embedding.

Later, we needed to be able to handle input of variable length. In order to do this, we added GlobalAveragePooling1D layer for global average pooling operation for temporal data.

Next layers which we appended were dense layers. The first one with bigger dimensionality of the output space and the second one with output dimension matching the number of the possible movie genres.

To avoid overfitting, we applied Dropout between two dense layers.

Because of lack of experience , we did not know how many layers will be the best for our project. That is way we started with two dense layers but while making experiments we decided that the neural network with only one dense layer is better in this case.

model.add(keras.layers.**Embedding**(input\_dim=vocabulary\_size,  
output\_dim= 512, input\_length=numberOfInputWords))

model.add(keras.layers.**GlobalAveragePooling1D**())

model.add(keras.layers.**Dense**(512, activation=tf.nn.tanh))

model.add(keras.layers.Dropout(0.3))

model.add(keras.layers.**Dense**(11, activation='softmax'))

* 1. Optimizer

Based on opinions in many articles about deep learning, we chose the Adam optimization algorithm as an optimizer in our project. The Adam is considered as one of the fastest algorithm.

* 1. Activation function

We made a lot of experiments to choose the best activation functions for dense layers. These two layers had different task to do, so we also matched different activation functions. Softmax  is frequently used in classifications – it turns numbers into probabilities. Pushing one result closer to 1 while another closer to 0, it sums outputs to 1 makes great probability analysis. That is way we choose it.

Another activation function was hyperbolic tangent which gave us the best results while testing different options.

1. Results

/\* tutaj te wykresy z repo, trzeba wybrać najsensowniejsze do analizy potem \*/

1. Conclusions